CSS:Cascading Style Sheets

CSS (Cascading Style Sheets) is used to style and layout web pages.

Types of CSS:
1.Inline
2.Internal
3.External

Which CSS overrides which one?
inline>Internal>External

```
selector {
  property: value;
}
```

Selector: The HTML element you want to style.
Property: The aspect of the element you want to change.
Value: The value of the property.

1. Basic Selectors

a. Universal Selector (*)

The universal selector selects all elements on a page.

```
* {
  margin: 0;
  padding: 0;
}
```

b. Type Selector

The type selector selects all elements of a given type (tag name).

```
p {
  color: blue;
}
```

c. Class Selector

The class selector selects all elements with a given class attribute. It is denoted by a period (`.`) followed by the class name.

```
.myClass {
  color: green;
}
```

d. ID Selector

The ID selector selects an element with a specific id attribute. It is denoted by a hash (`#`) followed by the id name. IDs should be unique within a page.

```
#myId {
  color: red;
}
```

e. Grouping Selectors

```
Grouping selectors apply the same styles to multiple elements.
```

```
h1, h2, h3 {
  color: green;
}
```

Properties:

1. Color and Background and Border

color: Sets the color of the text.

```
color: blue;
```

Border-radius,Border-width,border-style

background-color: Sets the background color of an element.

```
background-color: yellow;
```

-

background-image: Sets a background image for an element.
```
background-image: url('image.jpg');
```

-

background-repeat: Specifies whether the background image should repeat.
```
background-repeat: no-repeat;
```

-

background-size: Specifies the size of the background image.
```
background-size: cover;
```

-

background-position: Specifies the position of the background image.
```
background-position: center;
```

-

2. Text

font-family: Sets the font of the text.
```
font-family: Arial, sans-serif;
```

-

font-size: Sets the size of the font.
```
font-size: 16px;
```

-

font-weight: Sets the weight (boldness) of the font.
```
font-weight: bold;
```

-

font-style: Sets the style of the font (e.g., italic).
```
font-style: italic;
```

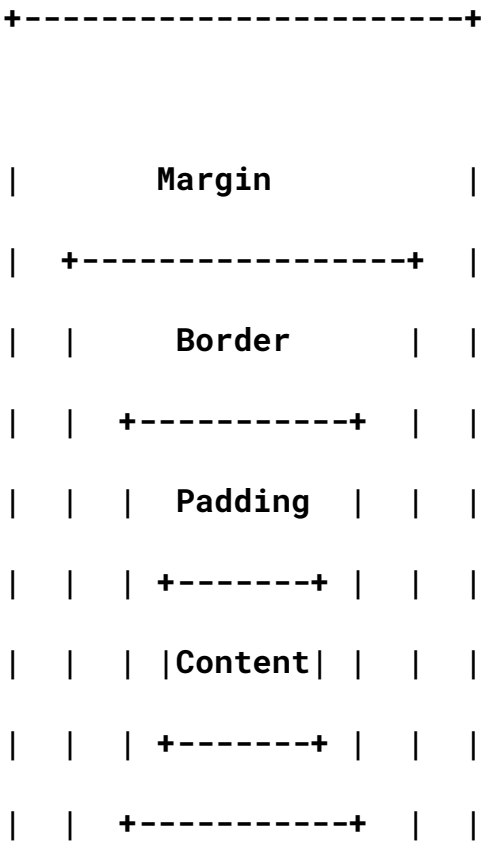
-

text-align: Sets the horizontal alignment of the text.
```
text-align: center;
```

-

text-decoration: Sets the decoration of the text (e.g., underline, line-through).
```
text-decoration: underline;
```

Box Model:



```
|  +-----------------+  |

+----------------------+
```

Components of the Box Model

Content: The actual content of the box, where text and images appear. The size of the content area can be controlled using properties like `width` and `height`.

```
width: 300px;

height: 200px;
```

1.

Padding: The space between the content and the border. Padding is inside the element and increases the element's total size. Padding can be set for all sides or individually for each side.

```
padding: 10px; /* All sides */

padding-top: 20px;

padding-right: 15px;

padding-bottom: 10px;

padding-left: 5px;
```

2.

Border: The line that surrounds the padding (if any) and content. Borders can be styled with various properties like `border-width`, `border-style`, and `border-color`.

```
border: 2px solid black; /* All sides */

border-top: 5px dotted red;

border-right: 3px dashed blue;

border-bottom: 2px solid green;
```

border-left: 1px double purple;

3.

Margin: The space outside the border, separating the element from other elements. Like padding, margins can be set for all sides or individually.

margin: 20px; /* All sides */

margin-top: 10px;

margin-right: 15px;

margin-bottom: 10px;

margin-left: 5px;

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CSS Box Model Example</title>
  <style>
    .box {
      width: 200px; /* Content width */
      height: 100px; /* Content height */
      padding: 20px; /* Padding */
      border: 5px solid black; /* Border */
      margin: 15px; /* Margin */
```

```
      background-color: lightblue; /* Background color for visualization */

    }



    .box-content-box {

      box-sizing: content-box; /* Default box-sizing */

    }



    .box-border-box {

      box-sizing: border-box; /* Includes padding and border in the element's
width and height */

    }

  </style>

</head>

<body>

  <h1>CSS Box Model Example</h1>

  <div class="box box-content-box">

    content-box: The width and height properties include only the content.
Padding and border are added outside of this.

  </div>

  <div class="box box-border-box">

    border-box: The width and height properties include content, padding, and
border. This makes the element's total size exactly 200px by 100px.

  </div>

</body>

</html>
```

Introduction to CSS Flexbox

Flexbox, short for Flexible Box Layout, is a CSS layout module that makes it easier to design flexible and responsive layouts. It is especially useful for creating complex layouts without relying on floats or positioning, which can be cumbersome and tricky to manage.

The Basics of Flexbox

Flexbox operates on two main axes:

1. Main Axis: The primary axis along which flex items are laid out. By default, this is horizontal (left to right).
2. Cross Axis: The axis perpendicular to the main axis. By default, this is vertical (top to bottom).

Flexbox has a container (the parent element) and flex items (the children of the container). The container controls the layout of the items using various flexbox properties.

Setting Up a Flexbox Container

To start using flexbox, you need to define a flex container by applying the `display: flex;` property to a parent element. Here's a basic example:

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Flexbox Basics</title>

    <style>

        .flex-container {

            display: flex;

            background-color: lightgray;
```

```
            padding: 10px;
        }

        .flex-item {
            background-color: steelblue;
            color: white;
            padding: 20px;
            margin: 10px;
            text-align: center;
        }
    </style>
</head>
<body>
    <div class="flex-container">
        <div class="flex-item">Item 1</div>
        <div class="flex-item">Item 2</div>
        <div class="flex-item">Item 3</div>
    </div>
</body>
</html>
```

Explanation

- Flex Container (`.flex-container`): This is the parent element that holds the flex items. Applying `display: flex;` makes it a flex container.

- Flex Items (.flex-item): These are the direct children of the flex container and automatically become flex items.

Important Flexbox Properties

1. **flex-direction: Defines the direction of the main axis.**
    - **row (default): Left to right.**
    - **row-reverse: Right to left.**
    - **column: Top to bottom.**
    - **column-reverse: Bottom to top.**

Example:

```
.flex-container {

    display: flex;

    flex-direction: column;

}
```

2.
3. **justify-content: Aligns items along the main axis.**
    - **flex-start (default): Items are aligned to the start of the container.**
    - **flex-end: Items are aligned to the end.**
    - **center: Items are centered.**
    - **space-between: Items are evenly distributed with space between them.**
    - **space-around: Items are evenly distributed with space around them.**

Example:

```
.flex-container {

    display: flex;

    justify-content: center;

}
```

4.
5. **align-items: Aligns items along the cross axis.**
    - **stretch (default): Items stretch to fill the container.**
    - **flex-start: Items are aligned to the start of the cross axis.**
    - **flex-end: Items are aligned to the end.**

- center: Items are centered along the cross axis.
- baseline: Items are aligned based on their text baseline.

Example:

```
.flex-container {

    display: flex;

    align-items: center;

}
```

6.
7. flex-wrap: Controls whether the items should wrap when they overflow the container.
   - nowrap (default): Items do not wrap.
   - wrap: Items wrap onto multiple lines.
   - wrap-reverse: Items wrap onto multiple lines in reverse order.

Example:

```
.flex-container {

    display: flex;

    flex-wrap: wrap;

8. }
```